```
sectors))/clm
        14955 LINKING/CLM
         2123 DEFECTIVE/CLM
        45557 CELL/CLM
        27664 CELLS/CLM
       308833 GROUP/CLM
         8152 SECTOR/CLM
        90292 GROUPS/CLM
         3907 SECTORS/CLM
L1          1 (LINKING (3A) DEFECTIVE (P) (CELL OR CELLS OR GROUP OR SECT
OR
              OR GROUPS OR SECTORS))/CLM

=> d kwic

US PAT NO:      5,297,148 [IMAGE AVAILABLE]              L1: 1 of 1

CLAIMS:

CLMS(1)

 We . . .

A memory card connectable to a computer system, comprising
 an array of electrically Erasable and Programmable Read Only Memory
  ("EEPROM")  cells  partitioned into a plurality of flash  sectors  ,
 each flash  sector  being a  group  of  cells  that are erasable
  together as a unit, and having a portion thereof reserved as redundant
  cells  ; and
 a memory controller for controlling operations of the EEPROM  cells  ,
 error detection means within said memory controller for detecting any
  defective  cells  within the array;
 defect pointers, each generated by said memory controller for
  linking  a detected  defective  cell 's address to that of a
  corresponding redundant  cell  substituting for the defective
  cells  , said defect pointer being stored within the array; and
 defective  cell  substituting means within said memory controller and
  responsive to said defect pointers for substituting said detected
  defective  cell  with said corresponding redundant  cells  .

=> Õ
=> s ((defect?) (P) (pointers or pointer or translating or pointing) (P) address
)/ab,clm and 371/10.?/ccls
         6196 DEFECT?/AB
         5882 DEFECT?/CLM
          476 POINTERS/AB
          970 POINTERS/CLM
         2026 POINTER/AB
         4721 POINTER/CLM
         1832 TRANSLATING/AB
         9666 TRANSLATING/CLM
         1327 POINTING/AB
         4546 POINTING/CLM
        11437 ADDRESS/AB
        25219 ADDRESS/CLM
           13 ((DEFECT?) (P) (POINTERS OR POINTER OR TRANSLATING OR POINT
ING
              ) (P) ADDRESS)/AB,CLM
          412 371/10.?/CCLS
L3          6 ((DEFECT?) (P) (POINTERS OR POINTER OR TRANSLATING OR POINT
ING
              ) (P) ADDRESS)/AB,CLM AND 371/10.?/CCLS

=> d 1-6

1.  5,359,570, Oct. 25, 1994, Solid state peripheral storage device;
Juei-Chi Hsu, et al., 365/230.01, 185, 200;  371/10.2  , 21.6 [IMAGE
AVAILABLE]

2.  5,297,148, Mar. 22, 1994, Flash eeprom system; Eliyahou Harari, et
al.,  371/10.2  ; 365/200;  371/10.1  ,  10.3  [IMAGE AVAILABLE]

3.  5,271,018, Dec. 14, 1993, Method and apparatus for media defect
management and media addressing; Litko Chan,  371/10.2  ; 360/48, 72.1;
369/32 [IMAGE AVAILABLE]

4.  5,200,959, Apr. 6, 1993, Device and method for defect handling in
semi-conductor memory; Stephen Gross, et al., 371/21.6; 364/236.2, 238,
240, 244, 244.6, 245.3, 251, 264, 264.5, 265, 280, 280.2, 282.1, 282.3,
285.3, DIG.1;  371/10.1  ,  10.2  , 11.1 [IMAGE AVAILABLE]

5.  4,525,839, Jun. 25, 1985, Method of controlling storage device;
Masafumi Nozawa, et al.,  371/10.2  , 40.3 [IMAGE AVAILABLE]

6.  4,380,066, Apr. 12, 1983, Defect tolerant memory; David H. Spencer,
et al.,  371/10.1  ; 364/927.8, 942, 942.04, 943.9, 943.91, 944, 944.2,
944.61, 944.92, 945.4, 954, 954.2, 957, 957.2, 959.1, 964, 964.7, 966.1,
966.5, 966.6, 970, 970.1, DIG.2; 365/200; 371/21.6 [IMAGE AVAILABLE]
```

=> ab 1, 3-6

US PAT NO: 5,359,570 [IMAGE AVAILABLE] L3: 1 of 6

ABSTRACT:
A solid state peripheral storage device receives an LSN address from a computer system and provides a mapping to a PSN data. The PSN data addresses memory units which are made out of solid state floating gate storage cells. In addition, a microsequencer controls the operation of the translation of the mapping of the LSN to PSN. Through the use of the mapping of LSN to PSN, defective sectors in the memory units can be mapped out and fresh, unused, defective-free sectors can then be replaced, all automatically, without user intervention. Finally, the microsequencer has error recovery routines to further enhance the reliability of the peripheral device.

US PAT NO: 5,271,018 [IMAGE AVAILABLE] L3: 3 of 6

ABSTRACT:
This invention provides a defect management scheme for mass storage devices such as disk drives. The data tracks formed on the surface of the storage media are divided into a plurality of zones. Each zone is divided into a number of logical partitions, with each partition containing a fixed number of sectors. Each partition also includes at least one local spare sector at the end of the partition. Each zone, which may consist of one or more partitions, includes a number of overflow spare sectors at the end of the zone. If there is a defective sector in a partition, the local spare sector is used to replace the defective sector. If there are more defective sectors in a partition than there are local spare sectors, an overflow spare sector is used to replace the additional defective sectors.

US PAT NO: 5,200,959 [IMAGE AVAILABLE] L3: 4 of 6

ABSTRACT:
A solid-state memory array such as an electrically erasable programmable read only memory (EEprom) or Flash EEprom array is used to store sequential data in a prescribed order. The memory includes a first information list containing addresses and defect types of previously detected defects. The defects are listed in the same prescribed order as that of the data. Only a simple controller is required to reference the information list so that writing or reading of the data will skip over the defective locations in the memory. New defects may be detected during writing by failure in verification, and those new defects will also be skipped. The memory also includes a second information list maintained by the controller. As data is written to the memory, addresses of file-markers and defects detected by write failure are entered into the list in the same prescribed order. This second list is referenced with the first list by the controller in subsequent reading to skip over both the previously and the newly detected defects.

US PAT NO: 4,525,839 [IMAGE AVAILABLE] L3: 5 of 6

ABSTRACT:
Disclosed is a control method for writing or reading sequentially a plurality of blocks of data for a storage device using a recording medium, such as a photo disk, with a number of recording blocks each having a unique address . In writing data, a plurality of data blocks to be written concurrently are given in their pointer fields the first address indicating the same alternative block, and these data are written in consecutive recording blocks on the recording medium. If blocks with write errors are detected, the data blocks corresponding to the error blocks are written sequentially in an alternative area starting from the first address , after replacing the pointer fields with the second address indicating the defective blocks, respectively. This allows the access of the alternative blocks for defective blocks by detecting the pointer address within the pointer fields of normal blocks preceding and following each of the defective blocks during the data reading operation.

US PAT NO: 4,380,066 [IMAGE AVAILABLE] L3: 6 of 6

ABSTRACT:
The invention is a defect tolerant memory for a computer system. The defect tolerant memory has a main memory, a redundant memory and a mask memory. The redundant memory receives and stores data redundant to that addressed to defective cells in the main memory. The redundant memory has multiple memory levels and uses a randomness technique to store redundant data for all chips of the main memory. The mask memory stores the location of each defect of main memory and indicates when a defective word is addressed in main memory. The mask memory is made up of multiple bit mask memories each cooperating with one of the redundant memory levels. Each bit-mask memory has multiple sub-memory units which use a randomness technique to store the addresses of defects in main memory.

Filing Date 4/24

3. 5,226,168, Jul. 6, 1993, **Semiconductor** **memory** configured to emulate floppy and hard **disk** magnetic storage based upon a determined storage capacity of the **semiconductor** **memory**; Junichi Kobayashi, et al., 395/800; 364/232.3, 236.2, 249, 280.2, 927.81, 952.1, 954, 975.2, DIG.1, DIG.2; 395/500 [IMAGE AVAILABLE]
=> d hit 3

US PAT NO: 5,226,168 [IMAGE AVAILABLE] L24: 3 of 3
TITLE: **Semiconductor** **memory** configured to emulate floppy and hard **disk** magnetic storage based upon a determined storage capacity of the **semiconductor** **memory**

ABSTRACT:
A compact, low power consumption, light weight, highly reliable and high speed information processing system is provided by employing a semiconductor **auxiliary** **storage** **device** in lieu of conventional magnetic storage or memory elements, such as floppy or hard disks. An access request to such a magnetic storage or memory element is converted to an access request for the semiconductor auxiliary storage which uses semiconductor integrated circuits without requiring any modification of existing programs, such as, application software and disk operating systems. A memory circuit in the semiconductor auxiliary storage comprises ROM and RAM, and a portion of the ROM contents is copied into the RAM so that access modification for programs and data is permitted while the basic program and data is retained in a nonvolatile manner.
Detd (6) EEPROM
CLAIMS:

CLMS(19)

19. A method for providing semiconductor type auxiliary storage for an information processing system having a CPU and a main program and data storage unit, the method comprising the steps of:
providing an auxiliary storage interface for transferring data between said CPU and auxiliary storage devices, said auxiliary storage interface being configured to access at least one auxiliary magnetic storage device;
connecting at least one **semiconductor** **memory** device to said auxiliary storage interface for receiving, storing, and retrieving data and programs;
providing a least a portion of a Basic Input/Output Operating system (BIOS) in communication with both said CPU and said auxiliary storage interface for controlling said **semiconductor** **memory** device so as to respond to said auxiliary storage interface in the same manner as said auxiliary magnetic storage device;
determining a storage capacity of said **semiconductor** **memory** device and outputting a first and second signal indicative of said determination;
detecting when said **semiconductor** **memory** device is to be treated as one of a floppy **disk** type and hard **disk** type magnetic storage, wherein upon receiving said first signal being outputted in said determining and outputting step, detecting that said **semiconductor** **memory** device is to be treated as said floppy **disk** type magnetic storage, and wherein upon receiving said second signal being outputted in said determining and outputting step,

```
L1          6 S ERASABLE TOGETHER
L2       9744 S (USER OR OVER HEAD OR OVERHEAD) (3A) DATA
L3        100 S (EEPROM OR EPROM OR FLASH OR FLOATING OR VOLATILE) (P) S
ECT
L4        108 S (EEPROM OR EPROM OR FLASH OR FLOATING OR VOLATILE) (P) S
ECT
L5     229563 S (DISK OR DISC)
L6         13 S L4 (P) L2
L7        554 S (OVERHEAD OR OVER HEAD) (2A) DATA
L8        225 S (OVERHEAD OR OVER HEAD) (1W) DATA
L9          1 S L4 AND L8
L10         0 S L4 AND ABCDEFG
L11        35 S (TOGETHER OR ONCE OR SIMULTANEOUS?) (P) L4
L12         1 S L7 AND L11
L13         6 S OVERHEAD (P) L4
=> d 1-6

1.  5,438,573, Aug. 1, 1995, Flash EEPROM array data and header file
structure; John S. Mangan, et al., 371/10.3, 2.1, 21.6 [IMAGE AVAILABLE]

2.  5,418,752, May 23, 1995, Flash EEPROM system with erase sector
select; Eliyahou Harari, et al., 365/218, 185.29 [IMAGE AVAILABLE]

3.  5,396,468, Mar. 7, 1995, Streamlined write operation for EEPROM
system; Eliyahou Harari, et al., 365/218, 185.01, 189.01 [IMAGE
AVAILABLE]

4.  5,369,615, Nov. 29, 1994, Method for optimum erasing of EEPROM;
Eliyahou Harari, et al., 365/185.19, 185.09, 185.14, 185.22, 185.31,
185.33 [IMAGE AVAILABLE]

5.  5,297,148, Mar. 22, 1994, Flash eeprom system; Eliyahou Harari, et
al., 371/10.2; 365/200; 371/10.3; 395/182.05 [IMAGE AVAILABLE]

6.  5,270,979, Dec. 14, 1993, Method for optimum erasing of EEPROM;
Eliyahou Harari, et al., 365/185.09, 185.11, 185.14, 185.33, 189.01, 200
[IMAGE AVAILABLE]
=> d hit 1-6

US PAT NO:     5,438,573 [IMAGE AVAILABLE]            L13: 1 of 6

ABSTRACT:
A file structure employed in a **flash** electrically **erasable** and
programmable read only **memory** ("**EEPROM**") system and aspects of
forming and using certain data fields within such a file structure. An
**array** of rows and columns of **EEPROM** **memory** **cells** is
divided into blocks of **cells** that are separately addressable for the
purpose of **erasing** an entire block of **cells** at the same time.
Each block contains several rows of **cells** with certain columns
thereof storing a **sector** of data, typically 512 bytes of data, and
other columns of **cells** within the same rows being used as spare
**cells** to replace any defective **sector** data **cells** and store
**overhead** (header) information about the block and the data
**sector**. Such **overhead** information includes pointers to locations
of any defective **sector** data **cells** within the block, whether the
block has been mapped out in favor of another block, error correction
codes for the **sector** data and the header information, and other
```

similar types of information.

DETDESC:

DETD(12)

The **memory** system of FIG. 1, in a preferred implementation, transfers data between the controller and **EEPROM** chip modules in eight byte chunks. Therefore, it is useful to show the organization of a block in terms of such chunks. With reference to FIG. 4, a **memory** block includes four rows of 18 chunks each. One chunk of each row at the same column address forms an **overhead** stack 215 and another vertically arranged group of chunks, one from each row, forms another stack 217. The remaining chunks shown unshaded in FIG. 4 hold the 512 bytes of **sector** data. By making the stacks 215 and 217 narrow, and by making them positionable away from each other in any of the chunk columns illustrated, it can be assured that at least the critical portions of the stacks are formed where there are no defective **EEPROM** **cells**. A primary characteristic of each of the blocks of the quadrant 203 (FIG. 3) is that all of the **memory** **cells** in that block are **erasable** at the same time by an **erase** pulse applied to a single addressed block.

US PAT NO: 5,418,752 [IMAGE AVAILABLE] L13: 2 of 6

DETDESC:

DETD(35)

FIG. 5 illustrates the **memory** architecture for the cell remapping scheme. As described before, the **Flash** **EEprom** **memory** is organized into **sectors** where the **cells** in each **sector** are **erasable** together. The **memory** architecture has a typical **sector** 401 organized into a data portion 403 and a spare (or shadow) portion 405. The data portion 403 is **memory** space available to the user. The spare portion 405 is further organized into an alternative defects data area 407, a defect map area 409, a header area 411 and an ECC and others area 413. These areas contain information that could be used by the controller to handle the defects and other **overhead** information such as headers and ECC.

US PAT NO: 5,396,468 [IMAGE AVAILABLE] L13: 3 of 6

DETDESC:

DETD(39)

FIG. 4a illustrates schematically a sample group of **cells**, N.sub.ref 261, of a **sector** 263. In general, N.sub.ref 261 may be assigned from any part of the **sector** 263. For example, in a 512 byte **flash** **sector**, consisting of 4 rows of 1024 **cells**, there will be 64 chunks of **cells**, with each chunk consisting of 64 **cells**. N.sub.ref may constitute one chunk of cell. If one chunk is considered insufficient, two or more chunks could be used at the cost of more **erase**-verify time **overhead**. To maintain generality the label N.sub.ref will be used to designate the reference chunks.

US PAT NO: 5,369,615 [IMAGE AVAILABLE] L13: 4 of 6

947, 947.6, 948.4, 948.5, 949, 952, 952.1, 959.1, 963, 963.3, 965, 965.5, 965.79, 968, DIG.1, DIG.2 [IMAGE AVAILABLE]

US PAT NO: 5,070,474 [IMAGE AVAILABLE] L3: 15 of 19
DATE FILED: Jul. 26, 1988

16. 5,062,042, Oct. 29, 1991, System for managing data which is accessible by file address or disk address via a disk track map; Joseph H. Binkley, et al., 395/600; 364/246.3, 254.8, 256.4, 261.2 [IMAGE AVAILABLE]

US PAT NO: 5,062,042 [IMAGE AVAILABLE] L3: 16 of 19
DATE FILED: Mar. 22, 1990

17. 4,953,122, Aug. 28, 1990, Pseudo-erasable and rewritable write-once optical disk memory system; Chris Williams, 395/404; 364/943.9, 943.91, 944, 952, 952.1, 952.31, 953, 953.1, 955, 955.5, 960, 961, 961.4, 963, 963.2, 968.1, 968.2, DIG.2; 369/53, 59; 395/412 [IMAGE AVAILABLE]

US PAT NO: 4,953,122 [IMAGE AVAILABLE] L3: 17 of 19
DATE FILED: Oct. 31, 1986

18. 4,787,031, Nov. 22, 1988, Computer with virtual machine mode and multiple protection rings; Paul A. Karger, et al., 395/800; 364/232.9, 259, 259.2, DIG.1 [IMAGE AVAILABLE]

US PAT NO: 4,787,031 [IMAGE AVAILABLE] L3: 18 of 19
DATE FILED: Jan. 4, 1985

19. 4,780,808, Oct. 25, 1988, Control of cache buffer for memory subsystem; Robert J. Moreno, et al., 395/800; 364/228.5, 232.3, 232.8, 236.2, 238.4, 239, 239.3, 239.7, 241.9, 243, 243.4, 245, 245.2, 245.3, 248.1, 249, 255.1, 255.7, DIG.1 [IMAGE AVAILABLE]

US PAT NO: 4,780,808 [IMAGE AVAILABLE] L3: 19 of 19
DATE FILED: Oct. 2, 1987
=> d kwic 15

US PAT NO: 5,070,474 [IMAGE AVAILABLE] L3: 15 of 19

SUMMARY:

BSUM(31)

In this embodiment, each track of data in the **disk** **emulator** may be comprised of thirty-two **sectors**, with each **sector** being comprised of sixty-four 66-bit words. This word length allows the **memory** interface timing to be very conservative while still maintaining a very fast transfer rate to the SMD **disk** controller. An additional benefit of the 66-bit word is the economy of parity. For a long word, the stored parity. . .

SUMMARY:

BSUM(32)

To further reduce the **memory** requirements for the DRAM array of the **disk** **emulator**, only the **sector**-specific data provided by the

SMD **disk** controller are stored in the DRAM array. In each **sector** on a **disk**, only the address field and the data field are unique. Accordingly, the **disk** **emulator** needs to store only address information and the data information and does so by making the zeroth word of the **sector** in the DRAM array the address field and the first through sixty-third words of the **sector** in DRAM array the data field. Since only the **sector**-specific data, the address field and the data field, are stored in the DRAM array, approximately 97% of the DRAM array is used for data storage while in a typical hard **disk** only 81% of the **disk** is available for data storage. Hence, the ability of the **disk** **emulator** to store only **sector**-specific data significantly enhances the utilization of the storage medium over prior art systems.

SUMMARY:

BSUM(59)

The **disk** **emulator** of this invention significantly improves both the seek time and the **sector** rotational latency. Also, the data storage medium in the **disk** **emulator** is used more efficiently than the data storage medium in a conventional hard **disk**. Finally, since the **disk** **emulator** has no mechanical or moving parts and since the novel error correction process corrects hard **memory** failures, the reliability of the **disk** **emulator** should be significantly better than the reliability of prior art hard **disk** drives.

DETDESC:

DETD(4)

In this embodiment, each track of data in the **disk** **emulator** is comprised of thirty-two **sectors** which are designated **sector** zero through **sector** thirty-one. Each **sector** is comprised of sixty-four 66-bit words (designated within the **sectors** as word zero through word sixty-three). The zeroth word in each **sector** is used to identify the **sector** and is the address for its respective **sector**. This word length allows the **memory** fetch timing to be very conservative while still maintaining a very fast transfer rate to the SMD **disk** controller. An additional benefit of the 66-bit word is the economy of parity. For a long word, the stored parity. . . small portion of the stored data. However, in view of the description of the present invention, the design of a **disk** **emulator** which utilizes a different word length and/or **sector** length will be apparent to those skilled in the art.

DETDESC:

DETD(7)

Since . . . control circuit 100 generated the lower order addresses, the location of the word that will initially be provided to SMD **disk** controller 107 is completely specified. Accordingly, the **disk** **emulator** fetches the specified 66-bit word plus 1 parity bit and provides that word to error correction circuit 103 over parallel bus 112. Error correction circuit 103, using the parity bit, analyzes the zeroth word (which is the address of **sector** zero) as described below, and if a bit of the 66-bit word changed while the word was stored, after storage

or during a read in the solid state **memory**, error correction circuit 103 corrects the hard error. If the error is not a hard error, error correction circuit 103, as described below, does not correct the error and so the error is passed to **disk** controller 107 which does correct the error.

DETDESC:

DETD(86)

The previous description of the **disk** **emulator** explained in general terms how a word is passed between latch circuit 803, shift register 802 and the SMD **disk** controller. However, the **disk** **emulator** must locate and address the **sector** of the track requested by the SMD **disk** controller. This is accomplished through ROM translation circuit 819, **memory** word counter circuit 807 and microprocessor 816 in conjunction with 8207 DRAM controller 820 and DRAM array 822 (FIG. 3).

DETDESC:

DETD(89)

Two . . . are available for translating the geometrical form of the address information to the binary structure suitable for addressing solid state **memory**. In the first method the portion of the computer operating system which interfaces with the **disk** **emulator**, i.e., the SMD **disk** controller, is modified. The modifications are made in the software **disk** driver that controls the **disk** **emulator**. In this method, the software **disk** driver is configured so that the driven **disk** has a binary number of heads, **sectors**, and cylinders. Thus, the information presented by the **disk** controller to the **disk** **emulator** describes contiguous binary addresses. Accordingly, in this method since the **disk** controller has been modified to generate a contiguous binary addresses, these addresses are simply used by the **disk** **emulator** to address the solid state **memory**. Hence, in this embodiment, ROM translator circuit 819 (not shown) is comprised of latches which capture the addresses provided by the SMD **disk** controller.

DETDESC:

DETD(168)

While . . . of the track requested by the SMD controller is now completely identified in the DRAM array, the counters in the **disk** **emulator** that are used to identify the **sector** and the words within the **sector** must be initialized. In **memory** word counter circuit 807 (FIG. 19), the low signal generated by NOR gate 495 in response to the high signal on **sector**/index line 712 loads counters 1054 and 1055 such that they count 64 clock pulses and then reset. In addition, the. . .

DETDESC:

DETD(176)

Thus, the index signal from microprocessor 1063 (FIG. 21) with the

sector address generated by microprocessor 1063 and the initialization of the counters in **memory** word counter circuit 807 (FIG. 19) has completed the definition of the address of the first word in DRAM array 822 that will be supplied to the SMD **disk** controller. This further demonstrates the access time advantage of the **disk** **emulator**. The initial track was located virtually instantaneously by the ROM translation circuit, and the zeroth word of the zeroth **sector**, which is the address field of the zeroth **sector**, is immediately identified as the first word which will be provided to the SMD controller. As is shown below, even if this is not the **sector** requested by the SMD controller, the **disk** **emulator** provides the correct **sector** to the SMD controller up to 500 times faster than a conventional hard **disk**.

DETDESC:

DETD(187)

The . . . RDB of the 8207 DRAM controller in FIG. 25 and as a result the 8207 DRAM controller initiates a read **memory** request for the zeroth word of the zeroth **sector** of the track specified by the SMD **disk** controller because, as previously described, the **disk** **emulator** has addressed the zero word of the zeroth **sector** in response to signals supplied by the SMD controller. As described previously, the zeroth word of the **sector** is the address field for the **sector**. The 8207 DRAM controller provides the address field for the zeroth **sector** on 66-bit parallel bus 700 and the stored parity bit for that address field on line D(0) from DRAM array. . .

DETDESC:

DETD(247)

During this period, since a **sector**/index pulse is not issued, there is no initialization of the circuits in the **disk** **emulator** and **memory** word counter circuit 807 (FIG. 19) and latch clock/error detection circuit 805 (FIG. 31) remain in the configuration created in supplying the address field to the SMD **disk** controller, as previously described. Accordingly, the state of the circuits in the **disk** **emulator** does not change until counter 1012 in second gap counter 809 (FIG. 16) rolls over.

DETDESC:

DETD(251)

Thus, at this point in the read from the **disk** **emulator**, the **disk** **emulator** has provided the complete **sector** prior to data field to the SMD **disk** controller even though the **sector**-specific address field was the only portion of the **sector** stored by the **disk** **emulator**. The string of zeros generated by the **disk** **emulator** to represent the first gap and the second gap are accepted as valid data by the SMD **disk** controller and since the **disk** **emulator** has no need for the nonsector-specific data, the generation of the zeroes has no affect on the ability of the **disk** **emulator** to interface with the SMD **disk** controller. Hence, the use of the volatile **memory** not only increases the speed of the **disk** **emulator** but also makes it possible to eliminate both the mechanical

mechanisms used in a conventional hard **disk** and the storage of the nonsector-specific information.

DETDESC:

DETD(266)

When the SMD **disk** controller asserts the write gate, the **disk** **emulator** is in a configuration identical to that described previously in the read cycle after the desired **sector** was identified and the read gate reasserted. The **memory** word counter circuit 807 (FIG. 19) is incremented to address the first word in the **sector**. The signal on the address zero detection complement line 754 from **memory** word counter circuit 807 disables address sync comparator 1122 (FIG. 14) and the signal on address one detection complement line 755 from **memory** word counter circuit 807 enables data sync comparator 1123 (FIG. 14). The second gap counter 809 (FIG. 16) is counting the bytes prior to the data field in the **sector** and read counter circuit 810 is generating zeros on output zero complement line 730.

DETDESC:

DETD(280)

The . . . data which is on 66-bit parallel bus 700 from shift register 802 (FIG. 10) in latch circuit 803. Hence, the **disk** **emulator**, which operates at 25 Megahertz or higher, has stored the first word in the data field so that it can be written to **memory** while the next word is being loaded into shift register 802. This permits operation at the high speed as well as writing to DRAM array 822 in a reasonable time frame. Also, this demonstrates how the **sector**-specific data field is detected and processed. Since the SMD **disk** **emulator** now passes the remainder of the data field to the **disk** **emulator**, the **disk** **emulator** must continue to generate a clock signal to latch circuit 803 on every 66th write clock pulse and write each. . .

DETDESC:

DETD(297)

This process continues until counter 1054 in **memory** word counter circuit 807 (FIG. 19) generates a carry pulse signal to AND gate 496 which in turn generates a. . . of the flip-flop 499, as previously described. On the next clock pulse on clock zero line 728 the signal on **sector** full line 757, which is connected to output terminal Q of flip-flop 498, goes high. The high signal on **sector** full line 757 drives the output signal from NOR gate 400 in write control circuit 806 (FIG. 15) low. The. . . of NOR gate 402 goes low and counters 1027, 1028 are in the clear mode, as previously described. Thus, the **disk** **emulator** reads the data field of the **sector** provided by the SMD **disk** controller and then inhibits write control circuit 806. Hence, the **disk** **emulator** functions in the write operation as a hard **disk** drive with substantially improved performance.
=>

US PAT NO: 5,070,474 [IMAGE AVAILABLE] L5: 12 of 15

DETDESC:

DETD(5)

When SMD **disk** controller 107 wants to read data from the **disk** **emulator** or write data to the **disk** **emulator**, SMD **disk** controller 107 sends a seek command to control circuit 100 over an on-cylinder line 121. SMD **disk** controller 107 simultaneously provides the cylinder and head address for the desired data over cylinder/head address bus 108 to address **translation** circuit 106. Address **translation** circuit 106 converts the signals provided by SMD **disk** controller 107 into the higher order **memory** addresses for the data the SMD **disk** controller wishes to read or write. These addresses are provided to **memory** address selector 105 over higher order address bus 109. Unlike a typical hard **disk** which must hunt for the specified track, the **disk** **emulator** determines the address of the desired track instantaneously using address **translation** circuit 106. Accordingly, address **translation** circuit 106 eliminates the head seek time associated with locating the specified track in a conventional hard **disk** drive.

DETDESC:

DETD(7)

Since the SMD controller 107 provided the cylinder and head address information, which was **translated** by address **translation** circuit 106 into the higher order addresses, and control circuit 100 generated the lower order addresses, the location of the word that will initially be provided to SMD **disk** controller 107 is completely specified. Accordingly, the **disk** **emulator** fetches the specified 66-bit word plus 1 parity bit and provides that word to error correction circuit 103 over parallel bus 112. Error correction circuit 103, using the parity bit, analyzes the zeroth word (which is the address of **sector** zero) as described below, and if a bit of the 66-bit word changed while the word was stored, after storage or during a read in the solid state **memory**, error correction circuit 103 corrects the hard error. If the error is not a hard error, error correction circuit 103, as described below, does not correct the error and so the error is passed to **disk** controller 107 which does correct the error.

DETDESC:

DETD(86)

The previous description of the **disk** **emulator** explained in general terms how a word is passed between latch circuit 803, shift register 802 and the SMD **disk** controller. However, the **disk** **emulator** must locate and address the **sector** of the track requested by the SMD **disk** controller. This is accomplished through ROM **translation** circuit 819, **memory** word counter circuit 807 and microprocessor 816 in conjunction with 8207 DRAM controller 820 and DRAM **array** 822 (FIG. 3).

DETDESC:

DETD(88)

When the SMD controller accesses a **disk**, the desired head and track data are usually provided as digital data. **Disk** controllers rarely provide contiguous binary addresses. Therefore, to interface the **disk** **emulator** with the SMD **disk** controller, the **disk** address information, the head and track data, provided by the SMD controller must be **translated** into a binary structure suitable for addressing a solid state **memory**.

DETDESC:

DETD(89)

Two different means are available for **translating** the geometrical form of the address information to the binary structure suitable for addressing solid state **memory**. In the first method the portion of the computer operating system which interfaces with the **disk** **emulator**, i.e., the SMD **disk** controller, is modified. The modifications are made in the software **disk** driver that controls the **disk** **emulator**. In this method, the software **disk** driver is configured so that the driven **disk** has a binary number of heads, **sectors**, and cylinders. Thus, the information presented by the **disk** controller to the **disk** **emulator** describes contiguous binary addresses. Accordingly, in this method since the **disk** controller has been modified to generate a contiguous binary addresses, these addresses are simply used by the **disk** **emulator** to address the solid state **memory**. Hence, in this embodiment, ROM **translator** circuit 819 (not shown) is comprised of latches which capture the addresses provided by the SMD **disk** controller.

DETDESC:

DETD(176)

Thus, the index signal from microprocessor 1063 (FIG. 21) with the **sector** address generated by microprocessor 1063 and the initialization of the counters in **memory** word counter circuit 807 (FIG. 19) has completed the definition of the address of the first word in DRAM **array** 822 that will be supplied to the SMD **disk** controller. This further demonstrates the access time advantage of the **disk** **emulator**. The initial track was located virtually instantaneously by the ROM **translation** circuit, and the zeroth word of the zeroth **sector**, which is the address field of the zeroth **sector**, is immediately identified as the first word which will be provided to the SMD controller. As is shown below, even if this is not the **sector** requested by the SMD controller, the **disk** **emulator** provides the correct **sector** to the SMD controller up to 500 times faster than a conventional hard **disk**.
=>

```
L1          160 S EMULAT? (P) MEMORY (P) DISK
L2        37032 S SECTOR#
L3           19 S L1 (P) L2
L4        11925 S (REMAP? OR TRANSLAT? OR PATCH?) (P) (MEMORY OR SECTOR# O
R A
L5           15 S L1 (P) L4
L6         6894 S (EEPROM OR FLASH OR FLOATING GATE) (P) MEMORY
L7          100 S L4 (P) L6
L8           11 S L7 (P) L2
L9            9 S L7 AND 371/CLAS
L10      159568 S DEFECT? OR UNUSABLE OR FAULT# OR UNCORRECTABLE
L11         441 S L4 (P) L10
L12           9 S L6 (P) L11
L13         157 S L4 AND L6 AND L10
L14           9 S L1 AND L13
=>
=> d l12 8-9 cit, fd ab kwic

8.  4,937,790, Jun. 26, 1990, Semiconductor memory device; Toshio Sasaki,
et al., 365/230.01, 189.01, 200; 371/21.1; 395/182.05 [IMAGE AVAILABLE]

US PAT NO:      4,937,790 [IMAGE AVAILABLE]            L12: 8 of 9
DATE FILED:     Aug. 3, 1988

ABSTRACT:
A semiconductor memory device is disclosed, in which a word line address
translation unit, a data line address translation unit, a first spare
memory and a second spare memory are provided in addition to a main
memory to relieve a defective memory cell in the main memory. Spare word
line address signals for selecting a spare word line on the first spare
memory are written in the word line address translation unit, spare data
line address signals for selecting a spare data line on the second spare
memory are written in the data line address translation unit, and each of
the word line address translation unit and the data line address
translation unit is constructed of an ordinary semiconductor memory of
the multi-bit output type.

SUMMARY:

BSUM(17)

 The word line address **translation** unit 7 for generating the spare
word line address signals 114 and the word line **fault** detection
signal 115 in response to the word line address signals 111 can be
constructed of an ordinary semiconductor **memory** of the multi-bit
output type, for example, one of non-volatile semiconductor memories such
as an EPROM (namely, electrically programmable read only **memory**), an
**EEPROM** (namely, electrically erasable and programmable read only
**memory**) and a fuse ROM (namely, fuse read only **memory**), or a
battery backuped semiconductor **memory** such as a battery backuped SRAM
(namely, battery backuped static random access **memory**). That is, the
word line address **translation** unit 7 does not include any associative
**memory**. Similarly, the data line address **translation** unit 8 for
generating the spare data line address signals 116 and the data line
**fault** detection signal 117 in response to the data line address
signals 110 can be constructed or an ordinary semiconductor **memory** of
the multi-bit output type, for example, one of non-volatile semiconductor
```

memories such as an EPROM, an **EEPROM** and a fuse ROM, or a battery backuped semiconductor **memory** such as a battery backuped RAM. That is, the data line address **translation** unit 8 includes no associative **memory**. Thus, according to the present invention, a **defect** on a main **memory** can be relieved in a relatively simple manner, and moreover hardware used is relatively simple in construction as shown in. . .

DETDESC:

DETD(12)

FIG. 3 shows a case where each of the address **translation** units 7 and 8 is constructed of an EPROM capable of writing information therein electrically and erasing information therefrom by ultraviolet rays. FIG. 4 shows a case where each of the address **translation** units 7 and 8 is constructed of an **EEPROM** capable of electrically writing information therein and electrically erasing information therefrom. FIG. 5 shows a case where each of the address **translation** units 7 and 8 is constructed of a battery backuped SRAM. Referring to FIG. 5, when the SRAM is disconnected. . . circuit 50 supplies a battery voltage V.sub.B to the SRAM, and thus the spare address signals 114 and 116 and **fault** detection signals 115 and 117 stored in the SRAM are prevented from vanishing. Further, each of the address **translation** units 7 and 8 may be constructed of a non-volatile **memory** such as a fuse ROM.

9. 4,802,119, Jan. 31, 1989, Single chip microcomputer with patching and configuration controlled by on-board non-volatile memory; Mark R. Heene, et al., 395/182.05; 364/925.6, 943.9, 944.92, 947, 947.2, 947.4, 965, 965.5, 965.76, 965.77, 965.79, 966.1, 966.4, 968.1, 968.3, 970, 970.1, DIG.2 [IMAGE AVAILABLE]

US PAT NO: 4,802,119 [IMAGE AVAILABLE] L12: 9 of 9
DATE FILED: Mar. 17, 1987

ABSTRACT:
A single chip microcomputer with **patching** and configuration is provided with blocks of **patch** **memory** which may be **patched** over **faulty** and/or obsolete areas of the microcomputer's **memory** map under control of starting address registers which are implemented in on-board non-volatile **memory**. The starting address registers, and enable registers which control whether each **patch** block is placed in the **memory** map, are programmable under control of the microcomputer's CPU. Newly programmed values in these registers are not effective to alter the **memory** map until a reset sequence enables a latch. In particular embodiments, **patch** blocks may overlie mask ROM, internal EPROM and/or **EEPROM**, external **memory** or devices or any other desireable portion of the **memory** map.

ABSTRACT:
A single chip microcomputer with **patching** and configuration is provided with blocks of **patch** **memory** which may be **patched** over **faulty** and/or obsolete areas of the microcomputer's **memory** map under control of starting address registers which are implemented in on-board non-volatile **memory**. The starting address registers, and enable registers which control whether each **patch** block is placed in the **memory** map, are programmable under control of the microcomputer's CPU. Newly programmed values in these registers are not effective to

alter the **memory** map until a reset sequence enables a latch. In particular embodiments, **patch** blocks may overlie mask ROM, internal EPROM and/or **EEPROM**, external **memory** or devices or any other desireable portion of the **memory** map.

SUMMARY:

BSUM(2)

The present invention relates, in general, to a single chip microcomputer (MCU) with **patching** and configuration controlled by on-board non-volatile **memory**. More specifically, the invention relates to a single chip MCU having certain registers implemented in electrically erasable programmable read-only **memory** (**EEPROM**) which are effective during power-on reset to supply information necessary to properly configure the MCU and to map blocks of **patch** **EEPROM** into the **memory** map to replace **defective** or obsolete portions of mask read-only **memory** (mask ROM), internal **EEPROM** or other elements of the normal **memory** map of the MCU.

DETDESC:

DETD(27)

For instance, any **patching** scheme according to the principles of the present invention must chose an appropriate size for each **patch** block. The smallest possible **patch** block size will normally be determined by the organization of the **EEPROM** **memory** used for **patching**. It may be, for instance, a single row or even one-half row of **EEPROM** cells (say, 32 or 16 bytes). Choosing a small **patch** block size provides great flexibility in **patching** small regions of **defective** or obsolete **memory**, but exacts penalties in that **patching** larger blocks of **memory** requires many **patch** blocks. The number of **patch** block registers required increases and each **patch** block register specifying a starting address must be longer.
=>

DEL HIS

L1 6 S ERASABLE TOGETHER

=> d 1-6

1. 5,428,621, Jun. 27, 1995, Latent defect handling in EEPROM devices; Sanjay Mehrotra, et al., 371/21.4, 10.3 [IMAGE AVAILABLE]

2. 5,418,752, May 23, 1995, Flash EEPROM system with erase sector select; Eliyahou Harari, et al., 365/218, 185.29 [IMAGE AVAILABLE]

3. 5,396,468, Mar. 7, 1995, Streamlined write operation for EEPROM system; Eliyahou Harari, et al., 365/218, 185.01, 189.01 [IMAGE AVAILABLE]

4. 5,369,615, Nov. 29, 1994, Method for optimum erasing of EEPROM; Eliyahou Harari, et al., 365/185.19, 185.09, 185.14, 185.22, 185.31, 185.33 [IMAGE AVAILABLE]

5. 5,297,148, Mar. 22, 1994, Flash eeprom system; Eliyahou Harari, et al., 371/10.2; 365/200; 371/10.3; 395/182.05 [IMAGE AVAILABLE]

6. 5,270,979, Dec. 14, 1993, Method for optimum erasing of EEPROM; Eliyahou Harari, et al., 365/185.09, 185.11, 185.14, 185.33, 189.01, 200 [IMAGE AVAILABLE]

=> d kwic 1-6

US PAT NO: 5,428,621 [IMAGE AVAILABLE] L1: 1 of 6

SUMMARY:

BSUM(32)

When . . . results in a column defect; and an erase line defect results in a sector (one or more rows that are **erasable** **together**) defect. Generally, the defective row, column or sector may be mapped out and replaced after data therein are recovered.

CLAIMS:

CLMS(9)

9. . . .
erase electrode;
the memory array is organized into sectors of cells consisting of one or more rows of cells that are **erasable** **together**; and
the access lines include erase lines connected to the erase electrodes of each sector of cells.

CLAIMS:

CLMS(22)

22. . . .
erase electrode;
the memory array is organized into sectors of cells consisting of one or more rows of cells that are **erasable** **together**; and
the access lines include erase lines connected to the erase electrodes

of each sector of cells.

CLAIMS:

CLMS(29)

29. . . . array being organized in a two-dimensional array and into sectors consisting of one or more rows of cells that are **erasable** **together**, the two-dimensional array being addressable by access lines, wherein one type of access line being word lines connected to the. . .

CLAIMS:

CLMS(40)

40. . . . array being organized in a two-dimensional array and into sectors consisting of one or more rows of cells that are **erasable** **together**, the two-dimensional array being addressable by access lines, wherein one type of access line being word lines connected to the. . .

CLAIMS:

CLMS(46)

46. . . . array being organized in a two-dimensional array and into sectors consisting of one or more rows of cells that are **erasable** **together**, the two-dimensional array being addressable by access lines, wherein one type of access line being word lines connected to the. . .

CLAIMS:

CLMS(48)

48. . . . array being organized in a two-dimensional array and into sectors consisting of one or more rows of cells that are **erasable** **together**, the two-dimensional array being addressable by access lines, wherein one type of access line being word lines connected to the. . .

CLAIMS:

CLMS(52)

52. . . . array being organized in a two-dimensional array and into sectors consisting of one or more rows of cells that are **erasable** **together**, the two-dimensional array being addressable by access lines, wherein one type of access line being word lines connected to the. . .

CLAIMS:

CLMS(55)

55. . . . array being organized in a two-dimensional array and into sectors consisting of one or more rows of cells that are **erasable**

together, the two-dimensional array being addressable by access lines, wherein one type of access line being word lines connected to the. . .

CLAIMS:

CLMS(58)

58. . . . array being organized in a two-dimensional array and into sectors consisting of one or more rows of cells that are **erasable** **together**, the two-dimensional array being addressable by access lines, wherein one type of access line being word lines connected to the. . .

US PAT NO: 5,418,752 [IMAGE AVAILABLE] L1: 2 of 6

DETDESC:

DETD(14)

In the present invention, the Flash EEprom memory is divided into sectors where all cells within each sector are **erasable** **together**. Each sector can be addressed separately and selected for erase. One important feature is the ability to select any combination. . .

DETDESC:

DETD(15)

FIG. . . . shown). The memory in each Flash EEprom chip is partitioned into sectors where all memory cells within a sector are **erasable** **together**. For example, each sector may have 512 byte (i.e. 512.times.8 cells) available to the user, and a chip may have. . .

DETDESC:

DETD(35)

FIG. . . . remapping scheme. As described before, the Flash EEprom memory is organized into sectors where the cells in each sector are **erasable** **together**. The memory architecture has a typical sector 401 organized into a data portion 403 and a spare (or shadow) portion. . .

DETDESC:

DETD(65)

In . . . memory array 33 is organized into sectors (typically 512 byte size) such that all memory cells within each sector are **erasable** **together**. Thus each sector may be considered to store a data file and a write operation on the memory array acts. . .

US PAT NO: 5,396,468 [IMAGE AVAILABLE] L1: 3 of 6

DETDESC:

DETD(12)

In the preferred embodiment, the array is partitioned into sectors where all cells within each sector are **erasable** **together**. For example, each sector may consist of 4 rows of cells having a total of 512 bytes (i.e., 512.times.8 cells). . .

US PAT NO: 5,369,615 [IMAGE AVAILABLE] L1: 4 of 6

DETDESC:

DETD(12)

In the preferred embodiment, the array is partitioned into sectors where all cells within each sector are **erasable** **together**. For example, each sector may consist of 4 rows of cells having a total of 512 bytes (i.e., 512.times.8 cells). . .

US PAT NO: 5,297,148 [IMAGE AVAILABLE] L1: 5 of 6

DETDESC:

DETD(14)

In the present invention, the Flash EEprom memory is divided into sectors where all cells within each sector are **erasable** **together**. Each sector can be addressed separately and selected for erase. One important feature is the ability to select any combination. . .

DETDESC:

DETD(15)

FIG. . . . shown). The memory in each Flash EEprom chip is partitioned into sectors where all memory cells within a sector are **erasable** **together**. For example, each sector may have 512 byte (i.e. 512.times.8 cells) available to the user, and a chip may have. . .

DETDESC:

DETD(35)

FIG. . . . remapping scheme. As described before, the Flash EEprom memory is organized into sectors where the cells in each sector are **erasable** **together**. The memory architecture has a typical sector 401 organized into a data portion 403 and a spare (or shadow) portion. . .

DETDESC:

DETD(65)

In . . . memory array 33 is organized into sectors (typically 512 byte size) such that all memory cells within each sector are **erasable** **together**. Thus each sector may be considered to store a data file and a write operation on the memory array acts. . .

CLAIMS:

CLMS(1)

We . . .
Only Memory ("EEPROM") cells partitioned into a plurality of flash sectors,
each flash sector being a group of cells that are **erasable** **together** as a unit, and having a portion thereof reserved as redundant cells; and
a memory controller for controlling operations of the. . .

US PAT NO: 5,270,979 [IMAGE AVAILABLE] L1: 6 of 6

DETDESC:

DETD(12)

In the preferred embodiment, the array is partitioned into sectors where all cells within each sector are **erasable** **together**. For example, each sector may consist of 4 rows of cells having a total of 512 bytes (i.e., 512.times.8 cells). . .
=>

```
L1          7816 S CARD (P) MEMORY
L2          4435 S ((OVERHEAD OR HEADER) (3A) (DATA OR INFORMATION))
L3            34 S L1 (P) L2
L4        495677 S (CONTROLLING OR CONTROLL? OR CONTROL###)/AB,CLM,TI
L5             1 S L3 (P) L4
L6            24 S L3 AND L4
=> d l5 ab, hit 1
```

US PAT NO: 5,428,685 [IMAGE AVAILABLE] L5: 1 of 1

ABSTRACT:

A method of protecting data in an IC memory card (11) having a ROM area (11a) and a RAM area (11b) and used as an external storage medium of a computer system includes storing key data for preventing copying in the ROM area and using the key data to prevent the copying of data stored in the RAM area. For example, when a file is written, second key data (k1, k2, . . .), which is inserted into the headers (H1, H2, . . .) of files (11c-1, 11c-2 . . .), is decided in such a manner that a fixed correlation will exist with respect to first key data (K1, K2, . . .) stored in the ROM area (11a), and the second key data is written in the RAM area (11b). When a file is read out of the IC memory card, a check is performed to determine whether the fixed correlation holds between the first key data that has been written in the ROM area and the second key data that has been inserting into the header of the file. The file is read out only if the fixed correlation holds.

CLAIMS:

CLMS(15)

15. IC **memory** **card** writing apparatus for protecting data recorded on an IC **memory** **card** used as an external storage medium of a computer system and having a first storage area and a second storage area, the apparatus comprising:
deciding means for deciding, when a file is written in the second storage area of the IC **memory** **card**, second key data, which is inserted into a header of the file, in such a manner that a fixed correlation is provided with respect to first key data stored in the first storage area of the IC **memory** **card**; and
write **control** means for writing the file in the second storage area after inserting the second key **data** into the **header** of the file.

CLAIMS:

CLMS(17)

17. IC **memory** **card** writing/reading apparatus for protecting data recorded on an IC **memory** **card** used as an external storage medium of a computer system and having a first storage area storing first key data and a second storage area, the apparatus comprising:
deciding means, for deciding when a file is written in the second storage area of the IC **memory** **card**, second key data, which is inserted into a header of the file, in such a manner that a fixed correlation is provided with respect to the first key data stored in the first storage area of the IC **memory** **card**;

write **control** means for writing the file in the second storage area after inserting the second key **data** into the **header** of the file;
checking means for checking, when a file is read out of the IC **memory** **card**, to determine whether the fixed correlation holds between the first key data that has been stored in the first storage area of the IC **memory** **card** and the second key data that has been inserted in the header of said file; and
read **control** means for reading out said file if the fixed correlation holds between the first and second key data, and not reading out said file if the fixed correlation does not hold.

CLAIMS:

CLMS(18)

18. IC **memory** **card** writing apparatus for protecting data recorded on an IC **memory** **card** used as an external storage medium of a computer system and having a first storage area and a second storage area, the apparatus comprising:
deciding means for deciding, when writing a file in the second storage area of the IC **memory** **card**, first and second key data in such a manner that a fixed correlation is provided therebetween; and
write **control** means for writing the first key data in the first storage area, inserting the second key **data** in a **header** of a file and writing the file in the second storage area.

CLAIMS:

CLMS(19)

19. IC **memory** **card** writing/reading apparatus for protecting data recorded on an IC **memory** **card** used as an external storage medium of a computer system and having first storage area and a second storage area, the apparatus comprising:
deciding means for deciding, when writing a file in the second storage area of the IC **memory** **card**, first and second key data in such a manner that a fixed correlation is provided therebetween;
write **control** means for writing the first key data in the first storage area, inserting the second key **data** in a **header** of a file and writing the file in the second storage area;
checking means for checking, when reading a file out of the IC **memory** **card**, to determine whether the fixed correlation holds between the first key data that has been written in the first storage area of the IC **memory** **card** and the second key data that has been inserted in the header of said file; and
read **control** means for reading out said file if the fixed correlation holds, and not reading out said file if the fixed correlation does not hold.
=>

60-178565 Sep. 12, 1985 L21: 1 of 3

AUXILIARY **STORAGE** **DEVICE**

INVENTOR: YUKIO WAYAMA, et al. (3)
ASSIGNEE: MEIDENSHA KK
APPL NO: 59-33802
DATE FILED: Feb. 24, 1984
PATENT ABSTRACTS OF JAPAN
ABS GRP NO: P426
ABS VOL NO: Vol. 10, No. 30
ABS PUB DATE: Feb. 5, 1986
INT-CL: G06F 13/16; G06F 3/06; G06F 12/02

ABSTRACT:

PURPOSE:To speed up an access time by using a semiconductor memory as a storage device and regarding a magnetic disk controller as an emulated memory controller.

CONSTITUTION:A central processing unit 1 is coupled with a memory controller 3 as the interface of a semiconductor disk device and information is transferred to and from the semiconductor memory unit 4 under the control of a main controller 3. The central processing unit 1 sends a request to transfer information to and from the controller 3 in the same access mode with the magnetic disk device, and the controller 3 performs processing required to write and read information in and out of the semiconductor memory unit 3 while processing necessary for the magnetic disk is omitted. The controller 3 **emulates** the controller of the magnetic disk device and also has a DMA and an interrupting function.
=>

detecting that said **semiconductor** **memory** device is to be treated as said hard **disk** type magnetic storage;

storing ID information comprising predetermined operating parameters for a plurality of magnetic storage formats corresponding to the type detected in said detecting step; and

initializing said **semiconductor** **memory** device using said BIOS so as to configure said **semiconductor** **memory** device in one of said plurality of storage formats based upon the type detected in said detecting step.

=>

L1 160 S EMULAT? (P) MEMORY (P) DISK
L2 37032 S SECTOR#
L3 19 S L1 (P) L2
=> d 1-19, cit, fd

1. 5,465,338, Nov. 7, 1995, Disk drive system interface architecture employing state machines; Donald W. Clay, 395/310, 404, 439, 894 [IMAGE AVAILABLE]

US PAT NO: 5,465,338 [IMAGE AVAILABLE] L3: 1 of 19
DATE FILED: Aug. 24, 1993

2. 5,463,765, Oct. 31, 1995, Disk array system, data writing method thereof, and fault recovering method; Hitoshi Kakuta, et al., 395/182.04; 371/40.1, 40.2, 40.4; 395/441 [IMAGE AVAILABLE]

US PAT NO: 5,463,765 [IMAGE AVAILABLE] L3: 2 of 19
DATE FILED: Mar. 18, 1993

3. 5,459,857, Oct. 17, 1995, Fault tolerant disk array data storage subsystem; Henry S. Ludlam, et al., 395/182.04; 364/268.5, DIG.1 [IMAGE AVAILABLE]

US PAT NO: 5,459,857 [IMAGE AVAILABLE] L3: 3 of 19
DATE FILED: Sep. 27, 1994

4. 5,459,850, Oct. 17, 1995, Flash solid state drive that emulates a disk drive and stores variable length and fixed lenth data blocks; Donald W. Clay, et al., 395/497.02; 364/245.1, 260.6, 260.7, DIG.1; 371/37.1; 395/442, 500 [IMAGE AVAILABLE]

US PAT NO: 5,459,850 [IMAGE AVAILABLE] L3: 4 of 19
DATE FILED: Feb. 19, 1993

5. 5,410,680, Apr. 25, 1995, Solid state memory device having serial input/output; Nagesh Challa, et al., 395/500; 364/236.2, 927.81, DIG.1, DIG.2 [IMAGE AVAILABLE]

US PAT NO: 5,410,680 [IMAGE AVAILABLE] L3: 5 of 19
DATE FILED: Nov. 17, 1993

6. 5,404,361, Apr. 4, 1995, Method and apparatus for ensuring data integrity in a dynamically mapped data storage subsystem; Anthony J. Casorso, et al., 371/40.1; 395/185.05 [IMAGE AVAILABLE]

US PAT NO: 5,404,361 [IMAGE AVAILABLE] L3: 6 of 19
DATE FILED: Jul. 27, 1992

7. 5,394,532, Feb. 28, 1995, Disk drive array memory system having instant format capability; Jay S. Belsan, 395/441; 364/243.2, DIG.1; 395/440 [IMAGE AVAILABLE]

US PAT NO: 5,394,532 [IMAGE AVAILABLE] L3: 7 of 19
DATE FILED: Apr. 15, 1992

8. 5,390,321, Feb. 14, 1995, General purpose parallel port interface; Ronald J. Proesel, 395/500; 364/235, 236.2, 927.92, 930, 952, 952.1,

DIG.1, DIG.2 [IMAGE AVAILABLE]

US PAT NO: 5,390,321 [IMAGE AVAILABLE] L3: 8 of 19
DATE FILED: Jul. 1, 1994

9. 5,335,338, Aug. 2, 1994, General purpose parallel port interface; Ronald J. Proesel, 395/500; 364/236.2, 248.1, DIG.1 [IMAGE AVAILABLE]

US PAT NO: 5,335,338 [IMAGE AVAILABLE] L3: 9 of 19
DATE FILED: May 31, 1991

10. 5,291,584, Mar. 1, 1994, Methods and apparatus for hard disk emulation; Nagesh Challa, et al., 395/500; 364/236.2, 927.81, DIG.1, DIG.2 [IMAGE AVAILABLE]

US PAT NO: 5,291,584 [IMAGE AVAILABLE] L3: 10 of 19
DATE FILED: Jul. 23, 1991

11. 5,283,884, Feb. 1, 1994, CKD channel with predictive track table; Jaishankar M. Menon, et al., 395/440; 364/232.3, 236.2, 238.3, 241.9, 243, 243.4, 243.41, 248.1, 251, 254, 254.3, 256.3, 259, 259.2, 260.6, 261, 265, 266.3, 270.5, DIG.1; 395/464, 600 [IMAGE AVAILABLE]

US PAT NO: 5,283,884 [IMAGE AVAILABLE] L3: 11 of 19
DATE FILED: Dec. 30, 1991

12. 5,218,691, Jun. 8, 1993, Disk emulation system; George B. Tuma, et al., 395/500; 364/221.2, 236.2, 916.3, 926.1, 926.5, 926.9, 926.92, 926.93, 927.81, 927.92, 927.93, 940, 940.1, 942, 942.7, 943.9, 944.92, 945.6, 947, 947.1, 947.2, 948.3, 949, 952, 952.1, 965, 965.5, DIG.1, DIG.2 [IMAGE AVAILABLE]

US PAT NO: 5,218,691 [IMAGE AVAILABLE] L3: 12 of 19
DATE FILED: Aug. 30, 1991

13. 5,193,184, Mar. 9, 1993, Deleted data file space release system for a dynamically mapped virtual data storage subsystem; Jay S. Belsan, et al., 395/404; 364/236.2, 246, 246.1, 246.3, 248.1, 256.3, 256.4, 268, 282.1, 282.3, DIG.1; 395/417, 441, 600 [IMAGE AVAILABLE]

US PAT NO: 5,193,184 [IMAGE AVAILABLE] L3: 13 of 19
DATE FILED: Jun. 18, 1990

14. 5,088,033, Feb. 11, 1992, Data processing system emulation in a window with a coprocessor and I/O emulation; Joseph H. Binkley, et al., 395/500; 364/222.81, 228.2, 228.6, 231, 232.3, 234, 236.2, 237.2, 237.3, 238, 238.3, 238.6, 238.9, 239, 239.6, 239.9, 240, 241.2, 241.3, 241.6, 242.3, 242.31, 242.6, 242.92, 244, 244.6, 247, 247.2, 247.8, 248.1, 256.3, 256.6, 259, 259.9, 260, 260.1, 261.3, 261.9, 262.4, 262.9, 264, 264.3, 265, 266.6, 270, 271, 271.2, 271.5, 280, 280.2, 286.1, 286.3, DIG.1 [IMAGE AVAILABLE]

US PAT NO: 5,088,033 [IMAGE AVAILABLE] L3: 14 of 19
DATE FILED: Mar. 23, 1990

15. 5,070,474, Dec. 3, 1991, Disk emulation system; George B. Tuma, et al., 395/500; 364/232.3, 236.2, 238.4, 238.6, 238.7, 248.1, 265, 267, 921.8, 927.8, 927.81, 927.92, 927.93, 933, 939, 940, 942, 943.9, 945,

DETDESC:

DETD(39)

FIG. 4a illustrates schematically a sample group of **cells**, N.sub.ref 261, of a **sector** 263. In general, N.sub.ref 261 may be assigned from any part of the **sector** 263. For example, in a 512 byte **flash** **sector**, consisting of 4 rows of 1024 **cells**, there will be 64 chunks of **cells** with each chunk consisting of 64 **cells**. N.sub.ref may constitute one chunk of cell. If one chunk is considered insufficient, two or more chunks could be used at the cost of more **erase**-verify time **overhead**. To maintain generality the label N.sub.ref will be used to designate the reference chunks.

US PAT NO: 5,297,148 [IMAGE AVAILABLE] L13: 5 of 6

DETDESC:

DETD(35)

FIG. 5 illustrates the **memory** architecture for the cell remapping scheme. As described before, the **Flash** **EEprom** **memory** is organized into **sectors** where the **cells** in each **sector** are **erasable** together. The **memory** architecture has a typical **sector** 401 organized into a data portion 403 and a spare (or shadow) portion 405. The data portion 403 is **memory** space available to the user. The spare portion 405 is further organized into an alternative defects data area 407, a defect map area 409, a header area 411 and an ECC and others area 413. These areas contain information that could be used by the controller to handle the defects and other **overhead** information such as headers and ECC.

US PAT NO: 5,270,979 [IMAGE AVAILABLE] L13: 6 of 6

DETDESC:

DETD(39)

FIG. 4a illustrates schematically a sample group of **cells**, N.sub.ref 261, of a **sector** 263. In general, N.sub.ref 261 may be assigned from any part of the **sector** 263. For example, in a 512 byte **flash** **sector**, consisting of 4 rows of 1024 **cells**, there will be 64 chunks of **cells**, with each chunk consisting of 64 **cells**. N.sub.ref may constitute one chunk of cell. If one chunk is considered insufficient, two or more chunks could be used at the cost of more **erase**-verify time **overhead**. To maintain generality the label N.sub.ref will be used to designate the reference chunks.
=>

=> d 1-16

1. 5,465,338, Nov. 7, 1995, Disk drive system interface architecture employing state machines; Donald W. Clay, 395/310, 404, 439, 894 [IMAGE AVAILABLE]

2. 5,418,752, May 23, 1995, Flash EEPROM system with erase sector select; Eliyahou Harari, et al., 365/218, 185.29 [IMAGE AVAILABLE]

3. 5,369,615, Nov. 29, 1994, Method for optimum erasing of EEPROM; Eliyahou Harari, et al., 365/185.19, 185.09, 185.14, 185.22, 185.31, 185.33 [IMAGE AVAILABLE]

4. 5,313,421, May 17, 1994, EEPROM with split gate source side injection; Daniel C. Guterman, et al., 365/185.15, 182, 184, 185.03, 185.16, 185.18, 185.19, 185.22, 185.3, 185.31, 218 [IMAGE AVAILABLE]

5. 5,297,148, Mar. 22, 1994, Flash eeprom system; Eliyahou Harari, et al., 371/10.2; 365/200; 371/10.3; 395/182.05 [IMAGE AVAILABLE]

6. 5,270,979, Dec. 14, 1993, Method for optimum erasing of EEPROM; Eliyahou Harari, et al., 365/185.09, 185.11, 185.14, 185.33, 189.01, 200 [IMAGE AVAILABLE]

7. 4,979,056, Dec. 18, 1990, Disk drive system controller architecture; John P. Squires, et al., 360/69, 70, 902, 904 [IMAGE AVAILABLE]

8. 4,979,055, Dec. 18, 1990, Disk drive system controller architecture utilizing embedded real-time diagnostic monitor; John P. Squires, et al., 360/69, 73.03, 78.04, 902, 904 [IMAGE AVAILABLE]

9. 4,656,475, Apr. 7, 1987, Method and apparatus for controlling distributed electrical loads; Edward B. Miller, et al., 340/825.06, 825.5 [IMAGE AVAILABLE]

10. 4,598,286, Jul. 1, 1986, Method and apparatus for controlling distributed electrical loads; Edward B. Miller, et al., 340/825.06, 825.18; 379/105 [IMAGE AVAILABLE]

11. 4,535,332, Aug. 13, 1985, Method and apparatus for controlling distributed electrical loads; Edward B. Miller, et al., 340/825.06, 825.5; 371/69.1 [IMAGE AVAILABLE]

12. 4,511,895, Apr. 16, 1985, Method and apparatus for controlling distributed electrical loads; Edward B. Miller, et al., 340/825.5, 825.52 [IMAGE AVAILABLE]

13. 4,489,385, Dec. 18, 1984, Method and apparatus for controlling distributed electrical loads; Edward B. Miller, et al., 364/493; 315/312; 324/103R; 364/571.03, 920.3, 923, 923.1, 926.9, 928, 929.1, 931, 931.3, 932.8, 935, 935.3, 937.01, 940, 940.81, 941, 941.6, 942.8, 943.9, 945.1, 948.4, 952, 952.4, 952.6, 965, 965.5, 967, 967.2, 970, 970.3, DIG.2 [IMAGE AVAILABLE]

14. 4,484,258, Nov. 20, 1984, Apparatus for controlling distributed electrical loads; Edward B. Miller, et al., 364/141; 340/825.07; 364/492 [IMAGE AVAILABLE]

15. 4,396,844, Aug. 2, 1983, Method and apparatus for controlling distributed electrical loads; Edward B. Miller, et al., 307/39, 40 [IMAGE AVAILABLE]

16. 4,367,414, Jan. 4, 1983, Method and apparatus for controlling distributed electrical loads; Edward B. Miller, et al., 307/38 [IMAGE AVAILABLE]
=> d kwic 6

US PAT NO: 5,270,979 [IMAGE AVAILABLE] L17: 6 of 16

ABSTRACT:
Various optimizing techniques are used for **erasing** semiconductor electrically **erasable** programmable read only memories (**EEPROM**). An **erase** algorithm accomplishes **erasing** of a group of **memory** **cells** by application of incremental **erase** pulses. Techniques include a 2-phase verification process interleaving between pulse applications; special handling of a sample of **cells** within each **erasable** unit group; defects handling; adaptive initial **erasing** voltages; and single-and hybrid-phase algorithms with **sector** to **sector** estimation of **erase** characteristics by table lookup. Techniques are also employed for **controlling** the uniformity of program/**erase** cycling of **cells** in each **erasable** unit group. Defects handling includes an adaptive data encoding scheme.

SUMMARY:

BSUM(8)

Optimized **erase** implementations for **EEPROM** systems have been disclosed in several copending U.S. patent applications. Copending U.S. patent application, Ser. No. 204,175, filed Jun. 8, 1988, by Dr. Eliyahou Harari, discloses an intelligent **erase** method for improved endurance. The **flash** **EEPROM** **cells** are **erased** by applying a pulse of **erasing** voltage followed by a read operation to verify if the **cells** are **erased** to the "**erased**" state. If not, further pulsing and verifying are repeated until the **cells** are verified to be **erased**. By **erasing** in this controlled manner the **cells** are not subjected to over-stress or over-erasure. Over-erasure tends to age the **EEPROM** device prematurely as well as to make the **cells** harder to re-program. Co-pending U.S. patent application, Ser. No. 337,566, filed Apr. 13, 1989, by Dr. Eliyahou Harari et al., discloses selective-multiple-**sector** **erase**, in which any combination of **flash** **sectors** may be **erased** together. Those **sectors** that have been verified as **erased** are removed from further pulses of **erasing** voltage, thereby preventing them from over-**erasing**.

SUMMARY:

BSUM(19)

According to one embodiment, a two-phase **erase** algorithm is applicable to an **array** of **EEPROM** **cells** that is organized in **erasable** unit of **sectors**. All **cells** in an addressable **sector** are to be **flash**-**erased** and a group of selected **sectors** are tagged for **erasing** together. A series of incrementing **erase**-voltage pulses are applied to the **cells** of all tagged

CLAIMS:

CLMS(5)

5. For an **array** of a plurality of electrically **erasable** and programmable read only **memory** **cells** having means for addressing the **cells** to program, read and **erase** their states, each cell having a field effect transistor that includes a **floating** gate and an **erase** gate, and having a natural threshold voltage that is alterable by programming or **erasing** to a level of charge on the **floating** gate to obtain an effective threshold voltage, wherein said natural threshold voltage corresponds to that when the **floating** gate has a level of charge equal to zero, said **array** being partitioned into **sectors** of **memory** **cells**, each **sector** being addressable for simultaneous **erasing** of all **cells** therein, a method of **erasing** a **sector** of addressed **cells** of the **array**, comprising the steps of:

in a first phase, pulsing an addressed **sector** with an **erase** voltage incremented successively from an initial **erase** voltage and verifying a sample of **cells** of the addressed **sector** in between pulses until more than a predetermined number of **cells** in the addressed **sectors** are completely **erased**; and thereafter

in a second phase, continuing pulsing the addressed **sector** with an **erase** voltage incremented from the last pulsing step and verifying all **cells** therein in between pulses until they are completely **erased**.

=> d kwic 2,3,5

US PAT NO: 5,418,752 [IMAGE AVAILABLE] L17: 2 of 16

ABSTRACT:
A system of **Flash** **EEprom** **memory** chips with **controlling** circuits serves as non-**volatile** **memory** such as that provided by magnetic disk drives. Improvements include selective multiple **sector** **erase**, in which any combinations of **Flash** **sectors** may be **erased** together. Selective **sectors** among the selected combination may also be de-selected during the **erase** operation. Another improvement is the ability to remap and replace defective **cells** with substitute **cells**. The remapping is performed automatically as soon as a defective cell is detected. When the number of defects in a **Flash** **sector** becomes large, the whole **sector** is remapped. Yet another improvement is the use of a write cache to reduce the number of writes to the **Flash** **EEprom** **memory**, thereby minimizing the stress to the device from undergoing too many write/**erase** cycling.

SUMMARY:

BSUM(14)

According to one aspect of the present invention, an **array** of **Flash** **EEprom** **cells** on a chip is organized into **sectors** such that all **cells** within each **sector** are **erasable** at once. A **Flash** **EEprom** **memory** system comprises one or more **Flash** **EEprom** chips under the control of a controller. The invention allows any combination of **sectors** among the chips to be selected and then **erased** simultaneously. This is faster and more efficient than prior art schemes where all the **sectors** must be **erased** every time or

sectors until the **cells** therein reach the "**erased**" state. The series of **erase** pulses steps from an initial voltage up to a predetermined, maximum allowable final voltage. The range between the two limits is set so that the optimum **erase** voltage of any typical **sectors** in the **array** will lie therein. After each pulse, the **cells** are read to verify if they have reached the **erased** state.

DETDESC:

DETD(32)

The . . . or exposed to much fewer cycles. This is quite a common occurrence in that the data pattern stored in a **memory** device invariably includes high activity addresses as well as addresses with zero or little activity. These bits are not truly. . . defective bits. Not only will this artificially cause the appearance of an errant tail, but it will result in large over-**erase** for the lightly written portion, since all **cells** in the **sectors** are **erased** together. The large over-**erase** will result in much higher net positive charge on a cell's **floating** gate than needed to establish the **erased** state.

DETDESC:

DETD(39)

FIG. 4a illustrates schematically a sample group of **cells**, N.sub.ref 261, of a **sector** 263. In general, N.sub.ref 261 may be assigned from any part of the **sector** 263. For example, in a 512 byte **flash** **sector**, consisting of 4 rows of 1024 **cells**, there will be 64 chunks of **cells**, with each chunk consisting of 64 **cells**. N.sub.ref may constitute one chunk of cell. If one chunk is considered insufficient, two or more chunks could be used at the cost of more **erase**-verify time **overhead**. To maintain generality the label N.sub.ref will be used to designate the reference chunks.

DETDESC:

DETD(53)

FIG. 7 outlines the main steps in the sequence of a two-phase **erase** algorithm incorporating the various aspects of the invention discussed above. Assume that an **array** of **EEPROM** **cells** partitioned into **flash** **sectors** is to be fully **erased**. Certain parameters established in conjunction with the **erase** algorithm are listed as follows:

DETDESC:

DETD(83)

The "stuck-at-0" type of defect can also be handled in similar fashion for typical **memory** devices. However, for **Flash** devices where a **sector** of **cells** are first **erased** simultaneously before any programming takes place, a two-pass operation is required to implement the inverse programming. That is, upon the detection of a "stuck-at-0" defect during programming operation in a **sector**, the whole **sector** must first be **erased** again before inverse programming can begin.

only one **sector** at a time can be **erased**. The invention further allows any combination of **sectors** selected for **erase** to be deselected and prevented from further **erasing** during the **erase** operation. This feature is important for stopping those **sectors** that are first to be **erased** correctly to the "**erased**" state from over **erasing**, thereby preventing unnecessary stress to the **Flash** **EEprom** device. The invention also allows a global de-select of all **sectors** in the system so that no **sectors** are selected for **erase**. This global reset can quickly put the system back to its initial state ready for selecting the next combination of **sectors** for **erase**. Another feature of the invention is that the selection is independent of the chip select signal which enables a particular chip for read or write operation. Therefore it is possible to perform an **erase** operation on some of the **Flash** **EEprom** chips while read and write operations may be performed on other chips not involved in the **erase** operation.

DRAWING DESC:

DRWD(4)

FIG. 2 is a schematic illustration of a system of **Flash** **EEprom** chips, among which **memory** **sectors** are selected to be **erased**;

DETDESC:

DETD(13)

In other devices such as Seeq Technology Incorporated's model 48512 **Flash** **EEprom** chip, the **memory** is divided into blocks (or **sectors**) that are each separately **erasable**, but only one at a time. By selecting the desired **sector** and going through the **erase** sequence the designated area is **erased**. While, the need for temporary **memory** is reduced, **erase** in various areas of the **memory** still requires a time consuming sequential approach.

DETDESC:

DETD(14)

In the present invention, the **Flash** **EEprom** **memory** is divided into **sectors** where all **cells** within each **sector** are **erasable** together. Each **sector** can be addressed separately and selected for **erase**. One important feature is the ability to select any combination of **sectors** for **erase** together. This will allow for a much faster system **erase** than by doing each one independently as in prior art.

DETDESC:

DETD(15)

FIG. 2 illustrates schematically selected multiple **sectors** for **erase**. A **Flash** **EEprom** system includes one or more **Flash** **EEprom** chips such as 201, 203, 205. They are in communication with a controller 31 through lines 209. Typically, the controller 31 is itself in communication with a microprocessor system (not shown). The **memory** in each **Flash** **EEprom** chip is partitioned into **sectors** where

all **memory** **cells** within a **sector** are **erasable** together. For example, each **sector** may have 512 byte (i.e. 512.times.8 **cells**) available to the user, and a chip may have 1024 **sectors**. Each **sector** is individually addressable, and may be selected, such as **sectors** 211, 213, 215, 217 in a multiple **sector** **erase**. As illustrated in FIG. 2, the selected **sectors** may be confined to one **EEprom** chip or be distributed among several chips in a system. The **sectors** that were selected will all be **erased** together. This capability will allow the **memory** and system of the present invention to operate much faster than the prior art architectures.

DETDESC:

DETD(33)

The nature of the **Flash** **EEprom** device predicates a higher rate of cell failure especially with increasing program/**erase** cycling. The hard errors that accumulate with use would eventually overwhelm the ECC and render the device unusable. One important. . . feature of the present invention is the ability for the system to correct for hard errors whenever they occur. Defective **cells** are detected by their failure to program or **erase** correctly. Also during read operation, defective **cells** are detected and located by the ECC. As soon as a defective cell is identified, the controller will apply defect mapping to replace the defective cell with a space cell located usually within the same **sector**. This dynamic correction of hard errors, in addition to conventional error correction schemes, significantly prolongs the life of the device.

DETDESC:

DETD(35)

FIG. 5 illustrates the **memory** architecture for the cell remapping scheme. As described before, the **Flash** **EEprom** **memory** is organized into **sectors** where the **cells** in each **sector** are **erasable** together. The **memory** architecture has a typical **sector** 401 organized into a data portion 403 and a spare (or shadow) portion 405. The data portion 403 is **memory** space available to the user. The spare portion 405 is further organized into an alternative defects data area 407, a. . . others area 413. These areas contain information that could be used by the controller to handle the defects and other **overhead** information such as headers and ECC.

DETDESC:

DETD(56)

Apart . . . are not present. In addition, the long synchronization times, sync mark detects and write gaps are not required. Thus the **overhead** needed for accessing the location where data is to be read or written is much less. All of these simplifications. . .

DETDESC:

DETD(65)

In the present invention, the **Flash** **EEprom** **memory** **array**

33 is organized into **sectors** (typically 512 byte size) such that all **memory** **cells** within each **sector** are **erasable** together. Thus each **sector** may be considered to store a data file and a write operation on the **memory** **array** acts on one or more such files.

CLAIMS:

CLMS(1)

We claim:

1. A **Flash** **EEprom** system comprising:
one or more integrated circuit chips each having an **array** of **Flash** **EEprom** **cells** partitioned into a plurality of **sectors**, each **sector** addressable for **erase** such that all **cells** therein are **erasable** simultaneously;
means for selecting a plurality of **sectors** among the one or more chips for **erase** operation;
means for simultaneously performing the **erase** operation on only the plurality of selected **sectors**; and
individual register associated with each **sector** for holding a status to indicate whether the **sector** is selected or not.

US PAT NO: 5,369,615 [IMAGE AVAILABLE] L17: 3 of 16

ABSTRACT:
Various optimizing techniques are used for **erasing** semiconductor electrically **erasable** programmable read only memories (**EEPROM**), An **erase** algorithm accomplishes **erasing** of a group of **memory** **cells** by application of incremental **erase** pulses, Techniques include a 2-phase verification process interleaving between pulse applications; special handling of a sample of **cells** within each **erasable** unit group; defects handling; adaptive initial **erasing** voltages; and single- and hybrid-phase algorithms with **sector** to **sector** estimation of **erase** characteristics by table lookup. Techniques are also employed for **controlling** the uniformity of program/**erase** cycling of **cells** in each **erasable** unit group, Defects handling includes an adaptive data encoding scheme.

SUMMARY:

BSUM(8)

Optimized **erase** implementations for **EEPROM** systems have been disclosed in several copending U.S. patent applications. Copending U.S. patent application, Ser. No. 204,175, filed Jun. 8, 1988, by Dr. Eliyahou Harari, discloses an intelligent **erase** method for improved endurance. The **flash** **EEPROM** **cells** are **erased** by applying a pulse of **erasing** voltage followed by a read operation to verify if the **cells** are **erased** to the "**erased**" state. If not, further pulsing and verifying are repeated until the **cells** are verified to be **erased**. By **erasing** in this controlled manner the **cells** are not subjected to over-stress or over-erasure. Over-erasure tends to age the **EEPROM** device prematurely as well as to make the **cells** harder to re-program. Co-pending U.S. patent application, Ser. No. 337,566, filed Apr. 13, 1989, by Dr. Eliyahou Harari et al., discloses selective-multiple-**sector** **erase**, in which any combination of **flash** **sectors** may be **erased** together. Those **sectors** that

have been verified as **erased** are removed from further pulses of **erasing** voltage, thereby preventing them from over-**erasing**.

SUMMARY:

BSUM(19)

According to one embodiment, a two-phase **erase** algorithm is applicable to an **array** of **EEPROM** **cells** that is organized in **erasable** unit of **sectors**. All **cells** if an addressable **sector** are to be **flash**-**erased** and a group of selected **sectors** are tagged for **erasing** together. A series of incrementing **erase**-voltage pulses are applied to the **cells** of all tagged **sectors** until the **cells** therein reach the "**erased**" state. The series of **erase** pulses steps from an initial voltage up to a predetermined, maximum allowable final voltage. The range between the two limits is set so that the optimum **erase** voltage of any typical **sectors** in the **array** will lie therein. After each pulse, the **cells** are read to verify if they have reached the **erased** state.

DETDESC:

DETD(32)

The . . . or exposed to much fewer cycles. This is quite a common occurrence in that the data pattern stored in a **memory** device invariably includes high activity addresses as well as addresses with zero or little activity. These bits are not truly. . . defective bits. Not only will this artificially cause the appearance of an errant tail, but it will result in large over-**erase** for the lightly written portion, since all **cells** in the **sectors** are **erased** together. The large over-**erase** will result in much higher net positive charge on a cell's **floating** gate than needed to establish the **erased** state.

DETDESC:

DETD(39)

FIG. 4a illustrates schematically a sample group of **cells**, N.sub.ref 261, of a **sector** 263. In general, N.sub.ref 261 may be assigned from any part of the **sector** 263. For example, in a 512 byte **flash** **sector**, consisting of 4 rows of 1024 **cells**, there will be 64 chunks of **cells** with each chunk consisting of 64 **cells**. N.sub.ref may constitute one chunk of cell. If one chunk is considered insufficient, two or more chunks could be used at the cost of more **erase**-verify time **overhead**. To maintain generality the label N.sub.ref will be used to designate the reference chunks.

DETDESC:

DETD(53)

FIG. 7 outlines the main steps in the sequence of a two-phase **erase** algorithm incorporating the various aspects of the invention discussed above. Assume that an **array** of **EEPROM** **cells** partitioned into **flash** **sectors** is to be fully **erased**. Certain parameters established in conjunction with the **erase** algorithm are listed as

follows: .

DETDESC:

DETD(139)

The "stuck-at-0" type of defect can also be handled in similar fashion for typical **memory** devices. However, for **Flash** devices where a **sector** of **cells** are first **erased** simultaneously before any programming takes place, a two-pass operation is required to implement the inverse programming. That is, upon the detection of a "stuck-at-0" defect during programming operation in a **sector**, the whole **sector** must first be **erased** again before inverse programming can begin.

CLAIMS:

CLMS(1)

What is claimed is:

1. For an **array** of a plurality of electrically **erasable** and programmable read only **memory** **cells** having means for addressing the **cells** to program, read and **erase** their states, each cell having a field effect transistor that includes a **floating** gate and an **erase** electrode, and having a natural threshold voltage that is alterable by programming or **erasing** to a level of charge on the **floating** gate to obtain an effective threshold voltage, wherein said natural threshold voltage corresponds to that when the **floating** gate has a level of charge equal to zero, said **array** being partitioned into **sectors** of **memory** **cells**, each **sector** being addressable for simultaneous **erasing** of all **cells** therein, a method of **erasing** a **sector** of addressed **cells** of the **array**, comprising the steps of:

reading a first set of **erase** parameters previously stored in said **sector**;

erasing said **sector** by using said first set of **erase** parameters;

determining a second set of **erase** parameter for optimally **erasing** of said **sector** in a subsequently **erase**; and

storing back said second set of **erase** parameters in said **sector**.

US PAT NO: 5,297,148 [IMAGE AVAILABLE] L17: 5 of 16

ABSTRACT:
A system of **Flash** **EEprom** **memory** chips with **controlling** circuits serves as non-**volatile** **memory** such as that provided by magnetic disk drives. Improvements include selective multiple **sector** **erase**, in which any combinations of **Flash** **sectors** may be **erased** together. Selective **sectors** among the selected combination may also be de-selected during the **erase** operation. Another improvement is the ability to remap and replace defective **cells** with substitute **cells**. The remapping is performed automatically as soon as a defective cell is detected. When the number of defects in a **Flash** **sector** becomes large, the whole **sector** is remapped. Yet another improvement is the use of a write cache to reduce the number of writes to the **Flash** **EEprom** **memory**, thereby minimizing the stress to the device from undergoing too many write/**erase** cycling.

SUMMARY:

BSUM(14)

According to one aspect of the present invention, an **array** of **Flash** **EEprom** **cells** on a chip is organized into **sectors** such that all **cells** within each **sector** are **erasable** at once. A **Flash** **EEprom** **memory** system comprises one or more **Flash** **EEprom** chips under the control of a controller. The invention allows any combination of **sectors** among the chips to be selected and then **erased** simultaneously. This is faster and more efficient than prior art schemes where all the **sectors** must be **erased** every time or only one **sector** at a time can be **erased**. The invention further allows any combination of **sectors** selected for **erase** to be deselected and prevented from further **erasing** during the **erase** operation. This feature is important for stopping those **sectors** that are first to be **erased** correctly to the "**erased**" state from over **erasing**, thereby preventing unnecessary stress to the **Flash** **EEprom** device. The invention also allows a global de-select of all **sectors** in the system so that no **sectors** are selected for **erase**. This global reset can quickly put the system back to its initial state ready for selecting the next combination of **sectors** for **erase**. Another feature of the invention is that the selection is independent of the chip select signal which enables a particular chip for read or write operation. Therefore it is possible to perform an **erase** operation on some of the **Flash** **EEprom** chips while read and write operations may be performed on other chips not involved in the **erase** operation.

DRAWING DESC:

DRWD(4)

FIG. 2 is a schematic illustration of a system of **Flash** **EEprom** chips, among which **memory** **sectors** are selected to be **erased**;

DETDESC:

DETD(13)

In other devices such as Seeq Technology Incorporated's model 48512 **Flash** **EEprom** chip, the **memory** is divided into blocks (or **sectors**) that are each separately **erasable**, but only one at a time. By selecting the desired **sector** and going through the **erase** sequence the designated area is **erased**. While, the need for temporary **memory** is reduced, **erase** in various areas of the **memory** still requires a time consuming sequential approach.

DETDESC:

DETD(14)

In the present invention, the **Flash** **EEprom** **memory** is divided into **sectors** where all **cells** within each **sector** are **erasable** together. Each **sector** can be addressed separately and selected for **erase**. One important feature is the ability to select any combination of **sectors** for **erase** together. This will allow for a much faster system **erase** than by doing each one independently

as in prior art.

DETDESC:

DETD(15)

FIG. 2 illustrates schematically selected multiple **sectors** for **erase**. A **Flash** **EEprom** system includes one or more **Flash** **EEprom** chips such as 201, 203, 205. They are in communication with a controller 31 through lines 209. Typically, the controller 31 is itself in communication with a microprocessor system (not shown). The **memory** in each **Flash** **EEprom** chip is partitioned into **sectors** where all **memory** **cells** within a **sector** are **erasable** together. For example, each **sector** may have 512 byte (i.e. 512.times.8 **cells**) available to the user, and a chip may have 1024 **sectors**. Each **sector** is individually addressable, and may be selected, such as **sectors** 211, 213, 215, 217 in a multiple **sector** **erase**. As illustrated in FIG. 2, the selected **sectors** may be confined to one **EEprom** chip or be distributed among several chips in a system. The **sectors** that were selected will all be **erased** together. This capability will allow the **memory** and system of the present invention to operate much faster than the prior art architectures.

DETDESC:

DETD(33)

The nature of the **Flash** **EEprom** device predicates a higher rate of cell failure especially with increasing program/**erase** cycling. The hard errors that accumulate with use would eventually overwhelm the ECC and render the device unusable. One important. . . feature of the present invention is the ability for the system to correct for hard errors whenever they occur. Defective **cells** are detected by their failure to program or **erase** correctly. Also during read operation, defective **cells** are detected and located by the ECC. As soon as a defective cell is identified, the controller will apply defect mapping to replace the defective cell with a space cell located usually within the same **sector**. This dynamic correction of hard errors, in addition to conventional error correction schemes, significantly prolongs the life of the device.

DETDESC:

DETD(35)

FIG. 5 illustrates the **memory** architecture for the cell remapping scheme. As described before, the **Flash** **EEprom** **memory** is organized into **sectors** where the **cells** in each **sector** are **erasable** together. The **memory** architecture has a typical **sector** 401 organized into a data portion 403 and a spare (or shadow) portion 405. The data portion 403 is **memory** space available to the user. The spare portion 405 is further organized into an alternative defects data area 407, a. . . others area 413. These areas contain information that could be used by the controller to handle the defects and other **overhead** information such as headers and ECC.

DETDESC:

DETD(56) .

Apart . . . are not present. In addition, the long synchronization times, sync mark detects and write gaps are not required. Thus the **overhead** needed for accessing the location where data is to be read or written is much less. All of these simplifications. . .

DETDESC:

DETD(65)

In the present invention, the **Flash** **EEprom** **memory** **array** 33 is organized into **sectors** (typically 512 byte size) such that all **memory** **cells** within each **sector** are **erasable** together. Thus each **sector** may be considered to store a data file and a write operation on the **memory** **array** acts on one or more such files.

CLAIMS:

CLMS(1)

We claim:

1. A **memory** card connectable to a computer system, comprising
an **array** of electrically **Erasable** and Programmable Read Only **Memory** ("**EEPROM**") **cells** partitioned into a plurality of **flash** **sectors**,
each **flash** **sector** being a group of **cells** that are **erasable** together as a unit, and having a portion thereof reserved as redundant **cells**; and
a **memory** controller for controlling operations of the **EEPROM** **cells**,
error detection means within said **memory** controller for detecting any defective **cells** within the **array**;
defect pointers, each generated by said **memory** controller for linking a detected defective cell's address to that of a corresponding redundant cell substituting for the defective **cells**, said defect pointer being stored within the **array**; and
defective cell substituting means within said **memory** controller and responsive to said defect pointers for substituting said detected defective cell with said corresponding redundant **cells**.
=>